Contribution ID: **88** Type: **not specified**

Efficient Userspace Optimistic Spinning Locks

Wednesday 11 September 2019 15:00 (45 minutes)

The most commonly used simple locking functions provided by the pthread library are pthread_mutex and pthread_rwlock. They are sleeping locks and so do suffer from unpredictable wakeup latency limiting locking throughput.

Userspace spinning locks can potentially offer better locking throughput, but they also suffer other drawbacks like lock holder preemption which will waste valuable CPU time for those lock spinning CPUs. Another spinning lock problem is contention on the lock cacheline when a large number of CPUs are spinning on it.

This talk presents a hybrid spinning/sleeping lock where a lock waiter can choose to spin in userspace or in the kernel waiting for the lock holder to release the lock. While spinning in the kernel, the lock waiters will queue up so that only the one at the queue head will be spinning on the lock reducing lock cacheline contention. If the lock holder is not running, the kernel lock waiters will go to sleep too so as not to waste valuable CPU cycles. The state of kernel lock spinners will be reflected in the value of lock. Thus userspace spinners can
monitor the lock state and determine the best way forward.

This new type of hybrid spinning/sleeping locks combine the best attributes of sleeping and spinning locks. It is especially useful for applications that need to run on large NUMA systems where potentially a large number of CPUs may be pounding on a given lock.

I agree to abide by the anti-harassment policy

Yes

Primary author: Mr LONG, Waiman (Red Hat)

Presenter: Mr LONG, Waiman (Red Hat)

Session Classification: LPC Refereed Track